Package ‘clickstream’

September 27, 2023

Type Package
Title Analyzes Clickstreams Based on Markov Chains
Version 1.3.3
Date 2023-09-27
Author Michael Scholz, Theo van Kraay
Maintainer Michael Scholz <michael.scholz@th-deg.de>
Encoding UTF-8
Description A set of tools to read, analyze and write lists of click sequences on websites (i.e., clickstream). A click can be represented by a number, character or string. Clickstreams can be modeled as zero- (only computes occurrence probabilities), first- or higher-order Markov chains.
License GPL-2
Depends R (>= 3.0.1), methods, igraph, stats, utils, reshape2, MASS
Imports plyr, Rsolnp, arules, linprog, ggplot2, ClickClust, parallel, data.table
LazyLoad yes
ByteCompile yes
RoxygenNote 7.2.3
NeedsCompilation no
Repository CRAN
Date/Publication 2023-09-27 14:50:02 UTC

R topics documented:
clickstream-package ........................................... 2
+,Pattern,Pattern-method .................................... 3
absorbingStates ............................................. 4
as.ClickClust ............................................... 5
as.clickstreams ........................................... 6
as.moltenTransactions .................................... 7
Analyzes Clickstreams Based on Markov Chains

Description

This package allows modeling clickstreams with Markov chains. It supports to model clickstreams as zero-order, first-order or higher-order Markov chains.

Details
Package: clickstream
Type: Package
Version: 1.3.3
Date: 2023-09-27
License: GPL-2
Depends: R (>= 3.0), methods

Author(s)
Michael Scholz <michael.scholz@th-deg.de>
Theo van Kraay <theo.vankraay@hotmail.com>

References

Examples

```r
# fitting a simple Markov chain and predicting the next click
clickstreams <- c("User1,h,c,c,p,c,h,c,p,p,c,p,p,o",
"User2,i,c,i,c,c,c,d",
"User3,h,i,c,i,c,p,c,c,p,c,i,d",
"User4,c,c,p,c,d",
"User5,h,c,c,p,c,p,p,i,p,o",
"User6,i,h,c,c,p,c,p,c,d")
cls <- as.clickstreams(clickstreams, header = TRUE)
mc <- fitMarkovChain(cls)
startPattern <- new("Pattern", sequence = c("h", "c"))
predict(mc, startPattern)
plot(mc)
```

### +,Pattern,Pattern-method

**Concatenates two Pattern objects**

Description

Concatenates two Pattern objects
### Usage

```r
## S4 method for signature 'Pattern,Pattern'
e1 + e2
```

#### Arguments

- `e1`: First pattern
- `e2`: Second pattern

#### Methods

```r
list("signature(e1 = "Pattern", e2 = "Pattern")")
```

Concatenates two `Pattern` objects.

### Description

Returns All Absorbing States

### Usage

```r
absorbingStates(object)
```

#### Arguments

- `object`: An instance of the `MarkovChain`-class

#### Methods

```r
list("signature(object = "MarkovChain")")
```

Returns the names of all states that never have a successor in a clickstream (i.e. that are absorbing).

### Author(s)

Michael Scholz `<michael.scholz@th-deg.de>`
Coerces a Clickstream Object to a ClickClust Object

Description

Coerces a Clickstream object to a ClickClust object.

Usage

as.ClickClust(clickstreamList)

Arguments

clickstreamList

A list of clickstreams.

Value

A list consisting of a dataset X and a vector of initial states y

Author(s)

Michael Scholz <michael.scholz@th-deg.de>

See Also

frequencies

Examples

clickstreams <- c("User1,h,c,c,p,c,h,c,p,p,c,p,p,o", "User2,i,c,i,c,c,c,d", "User3,h,i,c,i,c,p,c,c,p,c,i,d", "User4,c,c,p,c,d", "User5,h,c,c,p,c,p,p,i,p,o", "User6,i,h,c,c,p,p,c,p,c,d")
cls <- as.clickstreams(clickstreams, header = TRUE)
X <- as.ClickClust(cls)
as.clickstreams

Converts a character vector or a character list into a clickstream list.

Description

Converts a character vector or a character list into a clickstream list. Note that non-alphanumeric characters will be removed.

Usage

as.clickstreams(obj, sep = ",", header = TRUE)

Arguments

obj The character vector or character list which will be converted into a clickstream list. Each line of the vector must represent exactly one click stream.
sep The character separating clicks (default is ",").
header A logical flag indicating whether the first entry of each entry in the character vector is the name of the clickstream.

Value

A list of clickstreams. Each element is a vector of characters representing the clicks. The name of each list element is either extracted from the character vector or a unique number.

Author(s)

Michael Scholz <michael.scholz@th-deg.de>

See Also

print.Clickstreams, randomClickstreams

Examples

clickstreams <- c("User1,h,c,p,c,h,c,p,c,p,p,o", "User2,i,c,i,c,c,c,d", "User3,h,i,c,i,c,p,c,c,i,d", "User4,c,c,p,c,d", "User5,h,c,p,p,c,p,p,p,i,p,o", "User6,i,h,c,p,p,c,p,c,d")
cls <- as.clickstreams(clickstreams, header = TRUE)
print(cls)
as.moltenTransactions

Coerces a Clickstream Object to a Transactions Object

Description
Coerces a Clickstream object to a transactions object.

Usage
as.moltenTransactions(clickstreamList)

Arguments

clickstreamList
A list of clickstreams.

Value
An instance of the old class transactions

Author(s)
Michael Scholz <michael.scholz@th-deg.de>

See Also
frequencies

Examples

clickstreams <- c("User1,h,c,c,p,c,h,c,p,p,c,p,p,o", 
"User2,i,c,i,c,c,c,d", 
"User3,h,i,c,i,c,p,c,c,p,c,i,d", 
"User4,c,c,p,c,d", 
"User5,h,c,c,p,c,p,p,i,p,o", 
"User6,i,h,c,c,p,p,c,p,c,d")
c1s <- as.clickstreams(clickstreams, header = TRUE)
trans <- as.moltenTransactions(cls)
as.transactions  

Coerces a Clickstream Object to a Transactions Object

Description

Coerces a Clickstream object to a transactions object.

Usage

as.transactions(clickstreamList)

Arguments

clickstreamList  
A list of clickstreams.

Value

An instance of the class transactions

Author(s)

Michael Scholz <michael.scholz@th-deg.de>

See Also

frequencies

Examples

clickstreams <- c("User1,h,c,c,p,c,h,c,p,p,c,p,p,o",  
"User2,i,c,i,c,c,c,d",  
"User3,h,i,c,i,c,p,c,c,p,c,i,d",  
"User4,c,c,p,c,d",  
"User5,h,c,c,p,c,p,p,i,p,o",  
"User6,i,h,c,c,p,c,p,p,c,d")
cls <- as.clickstreams(clickstreams, header = TRUE)
trans <- as.transactions(cls)
### chiSquareTest

Calculates the chi-square statistic

**Description**

Calculates the chi-Square statistic, p-value, and degrees of freedom, for the first-order transition matrix of a MarkovChain object compared with observed state changes.

**Usage**

```r
chiSquareTest(cls, mc)
```

**Arguments**

- `cls`: The clickstream object.
- `mc`: The Markov chain against which to compare the clickstream data. Please note that the first-order transition matrix is used for performing the chi-square test.

**Author(s)**

Theo van Kraay <theo.vankraay@hotmail.com>

**Examples**

```r
clickstreams <- c("User1,h,c,p,c,h,c,p,p,c,p,p,o",
                   "User2,i,c,i,c,c,c,d",
                   "User3,h,i,c,i,c,p,c,p,c,c,i,d",
                   "User4,c,c,p,c,d")

csf <- tempfile()
writelines(clickstreams, csf)
cls <- readClickstreams(csf, header = TRUE)
unlink(csf)

mc <- fitMarkovChain(cls)
chiSquareTest(cls, mc)
```

---

### clusterClickstreams

Performs K-Means Clustering on a List of Clickstreams

**Description**

Performs k-means clustering on a list of clickstreams. For each clickstream a transition matrix of a given order is computed. These transition matrices are used as input for performing k-means clustering.

```r
clusterClickstreams
```
Usage

clusterClickstreams(clickstreamList, order = 0, centers, ...)

Arguments

  clickstreamList
  A list of clickstreams for which the cluster analysis is performed.

  order
  The order of the transition matrices used as input for clustering (default is 0; 0
  and 1 are possible).

  centers
  The number of clusters.

  ... Additional parameters for k-means clustering (see kmeans).

Value

This method returns a ClickstreamClusters object (S3-class). It is a list with the following
components:

  clusters The resulting list of Clickstreams objects.

  centers A matrix of cluster centres.

  states Vector of states

  totss The total sum of squares.

  withinss Vector of within-cluster sum of squares, one component per cluster.

  tot.withinss Total within-cluster sum of squares, i.e., sum(withinss).

  betweenss The between-cluster sum of squares, i.e., totss - tot.withinss.

Author(s)

Michael Scholz <michael.scholz@th-deg.de>

See Also

  print.ClickstreamClusters, summary.ClickstreamClusters

Examples

  clickstreams <- c("User1,h,c,c,p,c,h,c,p,p,c,p,o",
                     "User2,i,c,i,c,c,c,d",
                     "User3,h,i,c,i,c,p,c,c,i,d",
                     "User4,c,c,p,c,d",
                     "User5,h,c,c,p,p,c,p,p,i,p,o",
                     "User6,i,h,c,c,p,p,c,p,c,d")
  cls <- as.clickstreams(clickstreams, header = TRUE)
  clusters <- clusterClickstreams(cls, order = 0, centers = 2)
  print(clusters)
EvaluationResult-class

Class EvaluationResult

Description

Class EvaluationResult

Objects from the Class

Objects can be created by calls of the form new("EvaluationResult", ...). This S4 class describes EvaluationResult objects.

Author(s)

Michael Scholz <michael.scholz@th-deg.de>

See Also

mcEvaluate

Examples

# show EvaluationResult definition
showClass("EvaluationResult")

fitMarkovChain

Fits a List of Clickstreams to a Markov Chain

Description

This function fits a list of clickstreams to a Markov chain. Zero-order, first-order as well as higher-order Markov chains are supported. For estimating higher-order Markov chains this function solves the following linear or quadratic programming problem:

\[
\min \left\| \sum_{i=1}^{k} X - \lambda_i Q_i X \right\|
\]

s.t.

\[
\sum_{i=1}^{k} \lambda_i = 1
\]

\[
\lambda_i \geq 0
\]

The distribution of states is given as X. \( \lambda_i \) is the lag parameter for lag \( i \) and \( Q_i \) the transition matrix.
Usage

fitMarkovChain(clickstreamList, order = 1, verbose = TRUE, control = list())

Arguments

clickstreamList
A list of clickstreams for which a Markov chain is fitted.

order
(Optional) The order of the Markov chain that is fitted from the clickstreams. Per default, Markov chains with order=1 are fitted. It is also possible to fit zero-order Markov chains (order=0) and higher-order Markov chains.

verbose
(Optional) An optimal logical variable to indicate whether warnings and infos should be printed.

control
(Optional) The control list of optimization parameters. Parameter optimizer specifies the type of solver used to solve the given optimization problem. Possible values are "linear" (default) and "quadratic". Parameter use.lpsolve determines whether lpSolve or linprog is used as linear solver.

Details

For solving the quadratic programming problem of higher-order Markov chains, an augmented Lagrange multiplier method from the package Rsolnp is used.

Value

Returns a MarkovChain object.

Note

At least half of the clickstreams need to consist of as many clicks as the order of the Markov chain that should be fitted.

Author(s)

Michael Scholz <michael.scholz@th-deg.de>

References


See Also

MarkovChain, Rsolnp
Examples

# fitting a simple Markov chain

```r
clickstreams <- c("User1,h,c,c,p,c,h,c,p,p,c,p,p,o",
   "User2,i,c,i,c,c,d",
   "User3,i,c,i,c,p,c,c,p,c,i,d",
   "User4,c,c,p,c,d",
   "User5,h,c,c,p,c,p,p,i,p,o",
   "User6,i,h,c,c,p,c,p,c,d")

cls <- as.clickstreams(clickstreams, header = TRUE)

mc <- fitMarkovChain(cls)

show(mc)
```

fitMarkovChains

Generates a list of markov chains from a given set of clusters

Description

The purpose of this function is to generate pre-computed markov chain objects from clusters of clickstreams.

Usage

```r
fitMarkovChains(clusters, order = 1)
```

Arguments

- **clusters** The clusters from which to generate markov chain objects.
- **order** The order for the markov chain.

Author(s)

Theo van Kraay <theo.vankraay@hotmail.com>

Examples

```r
training <- c("User1,h,c,c,p,c,h,c,p,p,c,p,p,o",
   "User2,i,c,i,c,c,d",
   "User3,i,c,i,c,p,c,c,p,c,i,d",
   "User4,c,c,p,c,d")

test <- c("User1,h,c,c,p,p,h,c,p,p,c,p,p,o",
   "User2,i,c,i,c,c,d",
   "User4,c,c,c,c,d")

trainingCLS <- as.clickstreams(training, header = TRUE)

testCLS <- as.clickstreams(test, header = TRUE)
```
clusters <- clusterClickstreams(trainingCLS, centers = 2)
markovchains <- fitMarkovChains(clusters, order = 1)

---

**frequencies**

Generates a Data Frame of State Frequencies for All Clickstreams in a List of Clickstreams

### Description

Generates a data frame of state frequencies for all clickstreams in a list of clickstreams.

### Usage

```r
frequencies(clickstreamList)
```

### Arguments

- `clickstreamList`
  - A list of clickstreams.

### Value

- A data frame containing state frequencies for each clickstream.

### Author(s)

Michael Scholz <michael.scholz@th-deg.de>

### See Also

transactions

### Examples

```r
clickstreams <- c("User1,h,c,p,c,h,c,p,p,c,p,o", "User2,i,c,i,c,c,c,d", "User3,h,i,c,i,c,p,c,c,i,d", "User4,c,c,p,c,d", "User5,h,c,c,p,p,p,i,p,o", "User6,i,h,c,c,p,p,c,p,c,d")
cls <- as.clickstreams(clickstreams, header = TRUE)
frequencyDF <- frequencies(cls)
```
getConsensusClusters

Generates an optimal set of clusters for a clickstream object based on consensus clustering.

Description

This is an experimental function for a consensus clustering algorithm based on targeting a range of average next state probabilities derived when fitting each cluster to a markov chain.

Usage

getConsensusClusters(
    trainingCLS,
    testCLS,
    maxIterations = 5,
    optimalProbMean = 0.5,
    range = 0.3,
    centresMin = 2,
    clusterCentresRange = 0,
    order = 1,
    takeHighest = FALSE,
    verbose = FALSE
)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>trainingCLS</td>
<td>Clickstream object with training data (this should be the data used to build the markov chain object).</td>
</tr>
<tr>
<td>testCLS</td>
<td>Clickstream object with test data.</td>
</tr>
<tr>
<td>maxIterations</td>
<td>Number of times to iterate (repeat) through the k-means clustering.</td>
</tr>
<tr>
<td>optimalProbMean</td>
<td>The target average probability of each next page click prediction in a 1st order markov chain.</td>
</tr>
<tr>
<td>range</td>
<td>The range above the optimal probability to target.</td>
</tr>
<tr>
<td>centresMin</td>
<td>The minimum cluster centres to evaluate.</td>
</tr>
<tr>
<td>clusterCentresRange</td>
<td>the additional cluster centres to evaluate.</td>
</tr>
<tr>
<td>order</td>
<td>The order for markov chains that will be used to evaluate each cluster.</td>
</tr>
<tr>
<td>takeHighest</td>
<td>Determines whether to default to the highest mean next click probability, or error if the target is not reached after the given number of k-means iterations.</td>
</tr>
<tr>
<td>verbose</td>
<td>Should this function report extra information on progress?</td>
</tr>
</tbody>
</table>

Author(s)

Theo van Kraay <theo.vankraay@hotmail.com>
getConsensusClustersParallel

Generates an optimal set of clusters for a clickstream based on consensus clustering and with parallel computation

Description

This is an experimental function for a consensus clustering algorithm based on targeting a range of average next state probabilities derived when fitting each cluster to a markov chain. This function parallelizes k-means and fitToMarkovChain operations across computer cores, and depends on the parallel package to function.

Usage

getConsensusClustersParallel(
  trainingCLS,
  testCLS,
  maxIterations = 5,
  optimalProbMean = 0.5,
  range = 0.5,
  centresMin = 2,
  verbose = FALSE)

Examples

training <- c("User1,h,c,p,c,h,c,p,p,c,p,p,o",
               "User2,i,c,i,c,c,d",
               "User3,h,i,c,c,p,c,c,i,d",
               "User4,h,c,c,p,c,p,p,i,p,o",
               "User5,i,h,c,p,p,c,c,d",
               "User6,i,h,c,p,p,c,p,c,o",
               "User7,i,h,c,p,p,c,c,d",
               "User8,i,h,c,p,p,c,c,d,o")

test <- c(
    "User1,h,c,p,c,h,c,p,p,c,p,p,o",
    "User2,i,c,i,c,c,d",
    "User3,h,i,c,c,p,c,c,i,d"
  )

trainingCLS <- as.clickstreams(training, header = TRUE)

testCLS <- as.clickstreams(test, header = TRUE)

clusters <- getConsensusClusters(trainingCLS, testCLS, maxIterations=5,
                                   optimalProbMean=0.40, range = 0.70, centresMin = 2,
                                   clusterCentresRange = 0, order = 1, takeHighest = FALSE,
                                   verbose = FALSE)

markovchains <- fitMarkovChains(clusters)

startPattern <- new("Pattern", sequence = c("i", "h", "c", "p"))

mc <- getOptimalMarkovChain(startPattern, markovchains, clusters)
predict(mc, startPattern)
getConsensusClustersParallel

    clusterCentresRange = 0,
    order = 1,
    cores = 2,
    takeHighest = FALSE,
    verbose = FALSE
)

Arguments

trainingCLS      Clickstream object with training data (this should be the data used to build the
                 markov chain object).

testCLS          Clickstream object with test data.

maxIterations     Number of times to iterate (repeat) through the k-means clustering.

optimalProbMean   The target average probability of each next page click prediction in a 1st order
                 markov chain.

range             The range above the optimal probability to target.

centresMin        The minimum cluster centres to evaluate.

clusterCentresRange
                 the additional cluster centres to evaluate.

order             The order for markov chains that will be used to evaluate each cluster.

cores             Number of cores used for clustering.

takeHighest       Determines whether to default to the highest mean next click probability, or error
                 if the target is not reached after the given number of k-means iterations.

verbose           Should this function report extra information on progress?

Author(s)

Theo van Kraay <theo.vankraay@hotmail.com>

Examples

testing <- c("User1,h,c,p,c,h,c,p,c,p,p,o",
             "User2,c,c,c,d",
             "User3,i,c,p,c,h,c,p,c,i,d",
             "User4,h,c,c,p,p,p,p,i,p,o",
             "User5,i,h,c,c,h,c,p,c,p,c,d",
             "User6,i,h,c,c,p,c,p,c,o",
             "User7,i,h,c,c,p,c,p,c,d",
             "User8,i,h,c,c,p,c,p,c,d"
             )

test <- c(
    "User1,h,c,c,p,c,h,c,p,p,p,o",
    "User2,i,c,c,d",
    "User3,h,c,c,p,c,c,p,c,i,d"
    )

trainingCLS <- as.clickstreams(training, header = TRUE)
testCLS <- as.clickstreams(test, header = TRUE)

clusters <- getConsensusClustersParallel(trainingCLS, testCLS, maxIterations=3,
    optimalProbMean=0.40, range = 0.70, centresMin = 2,
    clusterCentresRange = 0, order = 1, cores = 1,
    takeHighest = FALSE, verbose = FALSE)

markovchains <- fitMarkovChains(clusters)
startPattern <- new("Pattern", sequence = c("i", "h", "c", "p"))
mc <- getOptimalMarkovChain(startPattern, markovchains, clusters)
predict(mc, startPattern)

getOptimalMarkovChain  Generates the optimal markov chains from a list of markov chains and corresponding clusters

Description

The purpose of this function is to predict from a pattern using pre-computed markov chains and corresponding clusters. The markov chain corresponding with the cluster that is the best fit to the prediction value is used.

Usage

generateMarkovChains(startPattern, markovchains, clusters)

Arguments

startPattern  The pattern object to be used.
markovchains  The pre-computed markov chains generated from a set of clusters.
clusters  The corresponding clusters (should be in the corresponding order as the markov chains).

Author(s)

Theo van Kraay <theo.vankraay@hotmail.com>

Examples

testing <- c("User1,h,c,p,c,h,c,p,c,p,p,o",
    "User2,i,c,i,c,c,d",
    "User3,h,i,c,i,c,p,c,c,i,d",
    "User4,c,c,c,d")

test <- c("User1,h,c,p,c,h,c,p,p,o",
    "User2,i,c,i,c,c,d",
    "User4,c,c,c,d")

trainingCLS <- as.clickstreams(training, header = TRUE)
testCLS <- as.clickstreams(test, header = TRUE)

clusters <- clusterClickstreams(trainingCLS, centers = 2)
markovchains <- fitMarkovChains(clusters, order = 1)
startPattern <- new("Pattern", sequence = c("c"))
mc <- getOptimalMarkovChain(startPattern, markovchains, clusters)
predict(mc, startPattern)

hmPlot

Plots a Heatmap

Description
Plots a Heatmap

Usage
hmPlot(
  object, 
  order = 1, 
  absorptionProbability = FALSE, 
  title = NA, 
  lowColor = "yellow", 
  highColor = "red", 
  flip = FALSE 
)

Arguments

object The MarkovChain for which a heatmap is plotted.
order Order of the transition matrix that should be plotted. Default is 1.
absorptionProbability Should the heatmap show absorption probabilities? Default is FALSE.
title Title of the heatmap.
lowColor Color for the lowest transition probability of 0. Default is "yellow".
highColor Color for the highest transition probability of 1. Default is "red".
flip Flip to horizontal plot. Default is FALSE.

Methods

list("signature(object = \"MarkovChain\")") Plots a heatmap for a specified transition matrix or the absorption probability matrix of a given MarkovChain object.

Author(s)

Michael Scholz <michael.scholz@th-deg.de>
initialize, Pattern-method

Description

Creates a new Pattern object

Usage

## S4 method for signature 'Pattern'
initialize(.Object, sequence, probability, absorbingProbabilities, ...)

Arguments

.Object      Pattern (name of the class)
sequence     Click sequence
probability  Probability for the click sequence
absorbingProbabilities
             Probabilities that the sequence will finally end in one of the absorbing states
...          Further arguments for the CallNextMethod function

Methods

list("signature(sequence = \"character\", probability = \"numeric\", absorbingProbabilities = \"numeric\")

Creates a new Pattern object.

Author(s)

Michael Scholz <michael.scholz@th-deg.de>
MarkovChain-class

**Description**

Class MarkovChain

**Objects from the Class**

Objects can be created by calls of the form `new("MarkovChain", ...`). This S4 class describes MarkovChain objects.

**Author(s)**

Michael Scholz <michael.scholz@th-deg.de>

**See Also**

`fitMarkovChain`

**Examples**

```r
# show MarkovChain definition
showClass("MarkovChain")

# fit a simple Markov chain from a list of click streams
clickstreams <- c("User1,h,c,c,p,c,h,c,p,p,c,p,p,o",
                  "User2,i,c,i,c,c,d",
                  "User3,h,i,c,i,c,p,c,c,p,c,i,d",
                  "User4,c,c,p,c,d",
                  "User5,h,c,c,p,c,p,p,i,p,o",
                  "User6,i,h,c,c,p,c,p,c,d")
clickstreams <- as.clickstreams(clickstreams, header = TRUE)
mc <- fitMarkovChain(cls)
show(mc)
```

**mcEvaluate**

`Evaluates the number of occurrences of predicted next clicks`

**Description**

Evaluates the number of occurrences of predicted next clicks vs. total number of starting pattern occurrences in a given clickstream. The predicted next click can be a markov chain of any order.
Usage

mcEvaluate(mc, startPattern, testCLS)

Arguments

mc
  a markovchain object (this should have been built from a set of training data)
startPattern
  the starting pattern we want to predict next click on, and evaluate observed occurrences in test data.
testCLS
  clickstream object with test data

Author(s)

Theo van Kraay <theo.vankraay@hotmail.com>

Examples

training <- c("User1,h,c,c,p,c,h,c,p,c,p,p,o",
              "User2,i,c,i,c,c,d",
              "User3,h,i,c,i,c,p,c,p,c,c,i,d",
              "User4,c,c,c,d")
test <- c("User1,h,h,h,h,c,c,p,p,h,c,p,p,o",
          "User2,i,c,i,c,c,d",
          "User4,c,c,c,d,c,c,c")

csf <- tempfile()
writeLines(training, csf)
trainingCLS <- readClickstreams(csf, header = TRUE)
unlink(csf)

csf <- tempfile()
writeLines(test, csf)
testCLS <- readClickstreams(csf, header = TRUE)
unlink(csf)

mc <- fitMarkovChain(trainingCLS, order = 1)
startPattern <- new("Pattern", sequence = c("c","c"))
res <- mcEvaluate(mc, startPattern, testCLS)
res

mcEvaluateAll

Evaluates all next page clicks in a clickstream training data set against a test data

Description

Evaluates all next page clicks in a clickstream training data set against a test data. Handles higher order by cycling through every possible pattern permutation. Produces a report of observed and expected values in a matrix.
Usage

mcEvaluateAll(
  mc,
  trainingCLS,
  testCLS,
  includeChiSquare = TRUE,
  returnChiSquareOnly = FALSE
)

Arguments

mc A markovchain object that corresponds to a list of clusters.
trainingCLS Clickstream object with training data (this should be the data used to build the markov chain object).
testCLS Clickstream object with test data.
includeChiSquare Should the result include the chi-square value?
returnChiSquareOnly Should the result only consist of the chi-square value?

Author(s)

Theo van Kraay <theo.vankraay@hotmail.com>

See Also

mcEvaluate

Examples

training <- c("User1,h,c,c,p,c,h,c,p,p,c,p,p", "User2,i,c,i,c,c,c,d")
test <- c("User1,h,c,c,p,c,h,c,d,p,c,d,p", "User2,i,c,i,p,c,c,d")
csf <- tempfile()
writeLines(training, csf)
trainingCLS <- readClickstreams(csf, header = TRUE)
unlink(csf)
csf <- tempfile()
writeLines(test, csf)
testCLS <- readClickstreams(csf, header = TRUE)
unlink(csf)
mc <- fitMarkovChain(trainingCLS, order = 2)
mcEvaluateAll(mc, trainingCLS, testCLS)
mcEvaluateAllClusters

Evaluates all next page clicks in a clickstream training data set against a test data on the basis of a set of pre-computed Markov chains and corresponding clusters. Handles higher order by cycling through every possible pattern permutation. Produces a report of observed and expected values in a matrix.

Usage

mcEvaluateAllClusters(
  markovchains,  
  clusters,  
  testCLS,  
  trainingCLS,  
  includeChiSquare = TRUE,  
  returnChiSquareOnly = FALSE
)

Arguments

markovchains  A list of MarkovChain-objects.
clusters      The list of clusters.
testCLS       Clickstream object with test data.
trainingCLS   Clickstream object with training data (this should be the data used to build the markov chain object).
includeChiSquare
              Should the result include the chi-square value?
returnChiSquareOnly
              Should the result only consist of the chi-square value?

Author(s)

Theo van Kraay <theo.vankraay@hotmail.com>

See Also

mcEvaluateAll
Examples

```r
training <- c("User1,h,c,c,p,c,h,c,h,o,p,p,c,p,p,o",
               "User2,i,c,i,c,c,o,o,o,i,d",
               "User3,h,i,c,i,c,o,i,p,c,c,p,c,c,i,d",
               "User4,c,c,p,c,d,o,i,h,o,o")

test <- c("User1,h,c,c,p,p,h,o,i,c,p,p,c,p,p,o",
           "User2,i,c,i,c,c,c,d",
           "User4,c,c,c,c,d")

csf <- tempfile()
writeLines(training, csf)
trainingCLS <- readClickstreams(csf, header = TRUE)
unlink(csf)

csf <- tempfile()
writeLines(test, csf)
testCLS <- readClickstreams(csf, header = TRUE)
unlink(csf)

clusters <- clusterClickstreams(trainingCLS, centers = 2, order = 1)
markovchains <- fitMarkovChains(clusters, order = 2)
mclinevaluateAllClusters(markovchains, clusters, testCLS, trainingCLS)
```

---

Pattern-class

<table>
<thead>
<tr>
<th>Class</th>
<th>Pattern</th>
</tr>
</thead>
</table>

Description

This S4 class describes a click pattern consisting of a sequence of clicks and a probability of occurrence.

Objects from the Class

Objects can be created by calls of the form `new("Pattern", sequence, probability, ...)`. This S4 class describes a click pattern consisting of a sequence of clicks and a probability of occurrence.

Author(s)

Michael Scholz <michael.scholz@th-deg.de>

See Also

`randomClicks`
Examples

```r
# show Pattern definition
showClass("Pattern")

# create simple Pattern objects
pattern1 <- new("Pattern", sequence = c("h", "c", "p"))
pattern2 <- new("Pattern", sequence = c("c", "p", "p"), probability = 0.2)
pattern3 <- new("Pattern", sequence = c("h", "p", "p"), probability = 0.35,
  absorbingProbabilities = data.frame(d = 0.6, o = 0.4))
```

---

### plot,MarkovChain-method

Plots a MarkovChain object

---

Description

Plots a MarkovChain object

Usage

```r
## S4 method for signature 'MarkovChain'
plot(x, order = 1, digits = 2, minProbability = 0, ...)
```

Arguments

- `x`: An instance of the MarkovChain-class
- `order`: The order of the transition matrix that should be plotted
- `digits`: The number of digits of the transition probabilities
- `minProbability`: Only transitions with a probability >= the specified minProbability will be shown
- `...`: Further parameters for the plot-function in package igraph

Methods

```r
list("
  signature(x = "MarkovChain", order = "numeric", digits = "numeric")")
```

Plots the transition matrix with order `order` of a MarkovChain object as graph.

Author(s)

Michael Scholz <michael.scholz@th-deg.de>
predict,MarkovChain-method

Predicts the Next Click(s) of a User

Description

Predicts the Next Click(s) of a User

Usage

## S4 method for signature 'MarkovChain'
predict(object, startPattern, dist = 1, ties = "random")

Arguments

- `object`: The MarkovChain used for predicting the next click(s)
- `startPattern`: Starting clicks of a user as Pattern object. A Pattern with an empty sequence is also possible.
- `dist` (Optional): The number of clicks that should be predicted (default is 1).
- `ties` (Optional): The strategy for handling ties in predicting the next click. Possible strategies are `random` (default) and `first`.

Methods

- `list("signature(object = \"MarkovChain\")")` This method predicts the next click(s) of a user. The first clicks of a user are given as Pattern object. The next click(s) are predicted based on the transition probabilities in the MarkovChain object. The probability distribution of the next click \( n \) is estimated as follows:

\[
X^{(n)} = B \cdot \sum_{i=1}^{k} \lambda_i Q_i X^{(n-i)}
\]

The distribution of states at time \( n \) is given as \( X^n \). The transition matrix for lag \( i \) is given as \( Q_i \). \( \lambda_i \) specifies the lag parameter and \( B \) the absorbing probability matrix.

Author(s)

Michael Scholz <michael.scholz@th-deg.de>

See Also

- `fitMarkovChain`
Examples

```r
# fitting a simple Markov chain and predicting the next click
clickstreams <- c("User1,h,c,p,c,h,c,p,p,c,p,p,o",
"User2,i,c,i,c,c,d",
"User3,h,i,c,i,c,p,c,c,i,d",
"User4,c,c,p,c,d",
"User5,h,c,c,p,p,c,p,p,i,p,o",
"User6,i,h,c,c,p,c,p,c,d")
class <- as.clickstreams(clickstreams, header = TRUE)
mc <- fitMarkovChain(class)
startPattern <- new("Pattern", sequence = c("h", "c"))
predict(mc, startPattern)

# predict with predefined absorbing probabilities
#
startPattern <- new("Pattern", sequence = c("h", "c"),
  absorbingProbabilities = data.frame(d = 0.2, o = 0.8))
predict(mc, startPattern)
```

predict.ClickstreamClusters

*Predicts the Cluster for a Given Pattern Object*

Description

Predicts the cluster for a given Pattern object. Potential clusters need to be identified with the method `clusterClickstreams` before predicting the cluster.

Usage

```r
## S3 method for class 'ClickstreamClusters'
predict(object, pattern, ...)  # predict.ClickstreamClusters
```

Arguments

- **object**: A `ClickstreamClusters` object containing the clusters. `ClickstreamClusters` represent the result of a cluster analysis on a list of clickstreams (see `clusterClickstreams`).
- **pattern**: Sequence of a user's initial clicks as Pattern object.
- **...**: Ignored parameters.

Value

Returns the index of the clusters to which the given Pattern object most probably belongs to.

Author(s)

Michael Scholz <michael.scholz@th-deg.de>
See Also

clusterClickstreams, print.ClickstreamClusters

Examples

clickstreams <- c("User1,h,c,p,c,h,c,p,p,c,p,p,o",
                  "User2,i,c,i,c,c,d",
                  "User3,h,i,c,i,p,c,c,p,c,c,i,d",
                  "User4,c,c,p,c,d",
                  "User5,h,c,c,p,p,p,p,i,p,o",
                  "User6,i,h,c,c,p,p,c,p,c,d")
cls <- as.clickstreams(clickstreams, header = TRUE)
clusters <- clusterClickstreams(cls, order = 0, centers = 2)
pattern <- new("Pattern", sequence = c("h", "c"))
predict(clusters, pattern)

print.ClickstreamClusters

Prints a ClickstreamClusters Object

Description

Prints a ClickstreamClusters object. A ClickstreamClusters object represents the result of a cluster analysis on a list of clickstreams (see clusterClickstreams).

Usage

## S3 method for class 'ClickstreamClusters'
print(x, ...)

Arguments

x A ClickstreamClusters object (see clusterClickstreams).

... Ignored parameters.

Author(s)

Michael Scholz <michael.scholz@th-deg.de>

See Also

clusterClickstreams, summary.ClickstreamClusters
Examples

```r
clickstreams <- c("User1,h,c,c,p,c,h,c,p,p,c,p,p,o",
                  "User2,i,c,i,c,c,c,d",
                  "User3,h,i,c,i,c,p,c,c,p,c,i,d",
                  "User4,c,c,p,c,d",
                  "User5,h,c,c,p,c,p,p,i,p,o",
                  "User6,i,h,c,c,p,c,p,c,d")
cls <- as.clickstreams(clickstreams, header = TRUE)
clusters <- clusterClickstreams(cls, order = 0, centers = 2)
print(clusters)
```

print.Clickstreams  Prints a Clickstreams Object

Description

Prints a Clickstreams object

Usage

```r
## S3 method for class 'Clickstreams'
print(x, ...)
```

Arguments

- **x**: A list of clickstreams.
- **...**: Ignored parameters.

Author(s)

Michael Scholz <michael.scholz@th-deg.de>

See Also

readClickstreams, randomClickstreams

Examples

```r
clickstreams <- c("User1,h,c,c,p,c,h,c,p,p,c,p,p,o",
                  "User2,i,c,i,c,c,c,d",
                  "User3,h,i,c,i,c,p,c,c,p,c,i,d",
                  "User4,c,c,p,c,d",
                  "User5,h,c,c,p,c,p,p,i,p,o",
                  "User6,i,h,c,c,p,c,p,c,d")
cls <- as.clickstreams(clickstreams, header = TRUE)
print(cls)
```
print.MarkovChainSummary

Prints the Summary of a MarkovChain Object

Description

Prints the summary of a MarkovChain object.

Usage

## S3 method for class 'MarkovChainSummary'
print(x, ...)

Arguments

x A MarkovChainSummary object generated with the function summary
...
Ignored parameters.

Author(s)

Michael Scholz <michael.scholz@th-deg.de>

See Also

summary

Examples

clickstreams <- c("User1,h,c,c,p,c,h,c,p,p,c,p,p,o",
"User2,i,c,i,c,c,c,d",
"User3,h,i,c,i,c,p,c,c,p,c,i,d",
"User4,c,c,p,c,d",
"User5,h,c,c,p,p,p,i,p,o",
"User6,i,h,c,c,p,p,c,p,c,d")
cls <- as.clickstreams(clickstreams, header = TRUE)
mc <- fitMarkovChain(cls)
print(summary(mc))
randomClicks  
Generates a Sequence of Clicks

Description

Generates a Sequence of Clicks

Usage

randomClicks(object, startPattern, dist)

Arguments

object  The MarkovChain used for generating the next click(s)
startPattern  Pattern containing the first clicks of a user. A Pattern object with an empty sequence is also possible.
dist  (Optional) The number of clicks that should be generated (default is 1).

Methods

list("signature(object = \"MarkovChain\")")  Generates a sequence of clicks by randomly walking through the transition graph of a given MarkovChain object.

Author(s)

Michael Scholz <michael.scholz@th-deg.de>

See Also

fitMarkovChain

Examples

# fitting a simple Markov chain and predicting the next click
clickstreams <- c("User1,h,c,p,c,h,c,p,p,c,p,p,o",
                   "User2,i,c,i,c,c,c,d",
                   "User3,h,i,c,i,c,p,c,c,i,d",
                   "User4,c,c,p,c,d",
                   "User5,h,c,c,p,p,c,p,p,i,p,o",
                   "User6,i,h,c,c,p,p,c,p,c,d")
clickstreams <- as.clickstreams(clickstreams, header = TRUE)
mc <- fitMarkovChain(clickstreams)
startPattern <- new("Pattern", sequence = c("h", "c"))
predict(mc, startPattern)
randomClickstreams

Generates a List of Clickstreams

Description

Generates a list of clickstreams by randomly walking through a given transition matrix.

Usage

randomClickstreams(
  states,
  startProbabilities,
  transitionMatrix,
  meanLength,
  n = 100
)

Arguments

states         Names of all possible states.
startProbabilities Start probabilities for all states.
transitionMatrix    Matrix of transition probabilities.
meanLength         Average length of the click streams.
n               Number of click streams to be generated.

Value

Returns a list of clickstreams.

Author(s)

Michael Scholz <michael.scholz@th-deg.de>

See Also

fitMarkovChain, readClickstreams, print.Clickstreams

Examples

# generate a simple list of click streams
states <- c("a", "b", "c")
startProbabilities <- c(0.2, 0.5, 0.3)
transitionMatrix <- matrix(c(0, 0.4, 0.6, 0.3, 0.1, 0.6, 0.2, 0.8, 0), nrow = 3)
cls <- randomClickstreams(states, startProbabilities, transitionMatrix, meanLength = 5, n = 10)
print(cls)
**readClickstreams**  
Reads a List of Clickstreams from File

**Description**
Reads a list of clickstream from a csv-file. Note that non-alphanumeric characters will be removed.

**Usage**
```
readClickstreams(file, sep = ",", header = FALSE)
```

**Arguments**
- `file` The name of the file which the clickstreams are to be read from. Each line of the file appears as one click stream. If it does not contain an absolute path, the file name is relative to the current working directory, `getwd`.
- `sep` The character separating clicks (default is "\,").
- `header` A logical flag indicating whether the first entry of each line in the file is the name of the clickstream user.

**Value**
A list of clickstreams. Each element is a vector of characters representing the clicks. The name of each list element is either the header of a clickstream file or a unique number.

**Author(s)**
Michael Scholz <michael.scholz@th-deg.de>

**See Also**
- `print.Clickstreams`
- `randomClickstreams`

**Examples**
```
clickstreams <- c("User1,h,c,c,p,c,h,c,p,p,c,p,p,o",
"User2,i,c,i,c,c,c,d",
"User3,h,i,c,i,c,p,c,c,p,c,c,i,d",
"User4,c,c,p,c,d",
"User5,h,c,c,p,p,c,p,p,i,p,o",
"User6,i,h,c,c,p,c,p,c,d")
csf <- tempfile()
writeLines(clickstreams, csf)
cls <- readClickstreams(csf, header = TRUE)
unlink(csf)
print(cls)
```
**Description**

Shows an `EvaluationResult` object

**Usage**

```r
## S4 method for signature 'EvaluationResult'
show(object)
```

**Arguments**

- `object` An instance of the `EvaluationResult`-class

**Methods**

- `list("signature(object = \"EvaluationResult\")")` Shows an `EvaluationResult` object.

**Author(s)**

Michael Scholz <michael.scholz@th-deg.de>

---

**Description**

Shows a `MarkovChain` object

**Usage**

```r
## S4 method for signature 'MarkovChain'
show(object)
```

**Arguments**

- `object` An instance of the `MarkovChain`-class

**Methods**

- `list("signature(object = \"MarkovChain\")")` Shows a `MarkovChain` object.
**Author(s)**
Michael Scholz <michael.scholz@th-deg.de>

---

**show,Pattern-method**  
*Shows a Pattern object*

---

**Description**
Shows a Pattern object

**Usage**
```r
## S4 method for signature 'Pattern'
show(object)
```

**Arguments**
- `object`  
  An instance of the Pattern-class

**Methods**
- `list("signature(object = \"Pattern\")")`  
  Shows a Pattern object.

---

**Author(s)**
Michael Scholz <michael.scholz@th-deg.de>

---

**states**  
*Returns All States*

---

**Description**
Returns All States

**Usage**
```r
states(object)
```

**Arguments**
- `object`  
  An instance of the MarkovChain-class

**Methods**
- `list("signature(object = \"MarkovChain\")")`  
  Returns the name of all states of a MarkovChain object.
Summary method: `summary.MarkovChain` is a method that prints the summary of a `MarkovChain` object.

**Description**

Prints the Summary of a MarkovChain Object

**Usage**

```r
## S4 method for signature 'MarkovChain'
summary(object)
```

**Arguments**

- **object**
  
  An instance of the `MarkovChain`-class

**Value**

Returns a `MarkovChainSummary` object.

- `list("desc")`  
  A short description of the MarkovChain object.
- `list("observations")`  
  The number of observations from which the MarkovChain has been fitted.
- `list("k")`  
  The number of estimation parameters.
- `list("logLikelihood")`  
  The maximal log-likelihood of the MarkovChain estimation.
- `list("aic")`  
  Akaike's Information Criterion for the MarkovChain object
- `list("bic")`  
  Bayesian Information Criterion for the MarkovChain object

**Methods**

- `list("signature(object = "MarkovChain")")`  
  Generates a summary for a given MarkovChain object

**Author(s)**

Michael Scholz <michael.scholz@th-deg.de>
summary.ClickstreamClusters

*Prints a Summary of a ClickstreamCluster Object*

**Description**

Prints a summary of a ClickstreamCluster object. A ClickstreamClusters object represents the result of a cluster analysis on a list of clickstreams (see `clusterClickstreams`).

**Usage**

```r
## S3 method for class 'ClickstreamClusters'
summary(object, ...) # S3 method for class 'ClickstreamClusters'
```

**Arguments**

- `object`: A ClickstreamClusters object returned by `clusterClickstreams`.
- `...`: Ignored parameters.

**Author(s)**

Michael Scholz <michael.scholz@th-deg.de>

**See Also**

`clusterClickstreams`, `print.ClickstreamClusters`

**Examples**

```r
clickstreams <- c("User1,h,c,c,p,c,h,c,p,p,p,p,o", "User2,i,c,i,c,c,c,d", "User3,h,i,c,i,c,p,c,c,p,c,i,d", "User4,c,c,p,c,d", "User5,h,c,c,p,c,p,p,p,i,p,o", "User6,i,h,c,c,p,p,c,p,c,d")
cls <- as.clickstreams(clickstreams, header = TRUE)
clusters <- clusterClickstreams(cls, order = 0, centers = 2)
summary(clusters)
```
summary.Clickstreams  *Prints a Summary of a Clickstreams Object*

**Description**

Prints a summary of a Clickstreams object.

**Usage**

```r
## S3 method for class 'Clickstreams'
summary(object, ...)  
```

**Arguments**

- `object` A Clickstreams object (see `readClickstreams`).
- `...` Ignored parameters.

**Author(s)**

Michael Scholz <michael.scholz@th-deg.de>

**See Also**

- `readClickstreams`, `randomClickstreams`

**Examples**

```r
clickstreams <- c("User1,h,c,p,c,h,c,p,p,c,p,p,o",
                  "User2,i,c,i,c,c,c,d",
                  "User3,h,i,c,i,c,p,c,c,c,i,d",
                  "User4,c,c,p,c,d",
                  "User5,h,c,c,p,c,p,p,i,p,o",
                  "User6,i,h,c,c,p,p,c,c,d")
cls <- as.clickstreams(clickstreams, header = TRUE)
summary(cls)
```
transientStates  

Returns All Transient States

Description

Returns All Transient States

Usage

transientStates(object)

Arguments

object  
An instance of the MarkovChain-class

Methods

list("signature(object = \"MarkovChain\")")  
Returns the names of all states that have a non-zero probability that a user will never return to them (i.e. that are transient).

Author(s)

Michael Scholz <michael.scholz@th-deg.de>

writeClickstreams  

Writes a List of Clickstreams to File

Description

Writes a list of clickstream to a csv-file.

Usage

writeClickstreams(
    clickstreamList,  
    file,  
    header = TRUE,  
    sep = ",",  
    quote = TRUE  
)
writeClickstreams

Arguments

- `clickstreamList`: The list of clickstreams to be written.
- `file`: The name of the file which the clickstreams are written to.
- `header`: A logical flag indicating whether the name of each clickstream element should be used as first element.
- `sep`: The character used to separate clicks (default is ",").
- `quote`: A logical flag indicating whether each element of a clickstream will be surrounded by double quotes (default is `TRUE`).

Author(s)

Michael Scholz <michael.scholz@th-deg.de>

See Also

`readClickstreams`, `clusterClickstreams`

Examples

```r
clickstreams <- c("User1,h,c,c,p,c,h,c,p,p,c,p,p,o",
                   "User2,i,c,i,c,c,d",
                   "User3,h,i,c,i,c,p,c,p,c,c,i,d",
                   "User4,c,c,p,c,d",
                   "User5,h,c,c,p,p,c,p,p,i,p,o",
                   "User6,i,h,c,c,p,p,c,c,d")
cls <- as.clickstreams(clickstreams, header = TRUE)
clusters <- clusterClickstreams(cls, order = 0, centers = 2)
writeClickstreams(cls, file = "clickstreams.csv", header = TRUE, sep = ",")

# Remove the clickstream file
unlink("clickstreams.csv")
```
Index

* **Markov**
  + clickstream-package, 2
* **chain**
  + clickstream-package, 2
* **classes**
  + EvaluationResult-class, 11
  + MarkovChain-class, 21
  + Pattern-class, 25
* **click**
  + clickstream-package, 2
* **methods**
  + absorbingStates, 4
  + fitMarkovChain, 11, 20, 21, 27, 32, 33
  + fitMarkovChains, 13
  + frequencies, 5, 7, 8, 14
  + getConsensusClusters, 15
  + getConsensusClustersParallel, 16
  + getOptimalMarkovChain, 18
  + getwd, 34
  + hmPlot, 19
  + hmPlot,MarkovChain-method (hmPlot), 19
  + initialize,Pattern-method, 20
  + kmeans, 10
  + MarkovChain, 12
  + MarkovChain-class, 21
  + mcEvaluate, 11, 21, 23
  + mcEvaluateAll, 22, 24
  + mcEvaluateAllClusters, 24
  + Pattern-class, 25
  + predict,MarkovChain-method, 26
  + predict,MarkovChain-method (predict), 27
  + predict.ClickstreamClusters, 28
  + print.ClickstreamClusters, 10, 29, 30, 38
  + print.ClickstreamClusters, 10, 29, 30, 38
  + print.MarkovChainSummary, 31
  + randomClicks, 25, 32
  + randomClicks,MarkovChain-method (randomClicks), 32
  + randomClicks,MarkovChain-method (randomClicks), 32
  + randomClickstreams, 6, 30, 33, 34, 39
  + readClickstreams, 30, 33, 34, 39, 41
  + Rsolnp, 12
  + show,EvaluationResult-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,MarkovChain-method, 35
  + show,Marko
INDEX

show, Pattern-method, 36
states, 36
states, MarkovChain-method (states), 36
summary, 31
summary, MarkovChain-method, 37
summary, ClickstreamClusters, 10, 29, 38
summary, Clickstreams, 39

transactions, 7, 8, 14
transientStates, 40
transientStates, MarkovChain-method (transientStates), 40

writeClickstreams, 40